EXAMINER'S AMENDMENT

1. An examiner's amendment to the record appears below. Should the changes and/or additions be unacceptable to applicant, an amendment may be filed as provided by 37 CFR 1.312. To ensure consideration of such an amendment, it MUST be submitted no later than the payment of the issue fee.

2. Authorization for this examiner's amendment was given in a telephone interview with Ramin Aghevli, Reg.# 43,462 on 11/30/2010.

3. This listing of claims will replace all prior versions and listings of claims in the application:

1. (Currently Amended) A method comprising:

interrupting a thread with an interrupt routine during a first critical region; and

setting the thread to restart at a beginning of the first critical region in response to an indication that the thread is working in a critical region, wherein the thread is to avoid disabling all interrupts before it enters the first critical region and avoid re-enabling the interrupts after exiting the first critical region,

wherein setting the thread to start at the beginning of the first critical region comprises setting a stack pointer and program counter so that the interrupted thread restarts at the beginning of the first critical region, incrementing a count of a number of

times the first critical region has been interrupted in response to the first critical region having been previously interrupted, saving program stack conditions of the interrupted thread in response to the thread not having been interrupted during the first critical region more than a maximum permitted number of times, setting a flag to indicate that a critical region is entered, performing critical region interrupt recovery work in response to an indication that the first critical region was previously interrupted, and attempting to complete first critical region work.

2-4. (Canceled)

5. (Currently Amended) The method of Claim [[4]] 1, wherein the critical region interrupt recovery work corrects errors that arise from previously interrupting the first critical region prior to its completion.

6. (Currently Amended) The method of Claim [[4]] 1, wherein the first critical region further includes the acts of:

setting a flag to indicate that critical region work is not being performed in response to the critical region work completing; and

setting to zero the count of the number of times the first critical region was interrupted.

7. (Original) The method of Claim 1 further comprising:

marking the thread as having been interrupted during the first critical region.

8. (Original) The method of Claim 1, further comprising indicating that the thread is not operating in a critical region.

9. (Currently Amended) A method comprising:

interrupting a thread during a first critical region;

setting the thread to restart at a beginning of the first critical region in response to an indication that the thread is working in a critical region, wherein setting the thread to start at the beginning of the first critical region comprises setting a stack pointer and program counter so that the interrupted thread restarts at the beginning of the first critical region;

incrementing a count of a number of times the first critical region has been interrupted in response to the first critical region having been previously interrupted; ~~and~~

saving program stack conditions of the interrupted thread in response to the thread not having been interrupted during the first critical region more than a maximum permitted number of times, wherein the thread is to avoid disabling all interrupts before it enters the first critical region and avoid re-enabling the interrupts after exiting the first critical region;

setting a flag to indicate that a critical region is entered;

performing critical region interrupt recovery work in response to an indication that the first critical region was previously interrupted;

clearing a flag indicating the first critical region was previously interrupted, wherein the critical region interrupt recovery work corrects errors that arise from previously interrupting the first critical region prior to its completion; and

commencing work in the first critical region.

10-12. (Canceled)

13. (Currently Amended) The method of Claim [[12]] 9, further comprising:

setting a flag to indicate that critical region work is not being performed in response to the critical region work completing; and

resetting a counter of times the first critical region was interrupted to zero.

14. (Previously Presented) The method of Claim 9, further comprising:

~~setting a flag to indicate that a critical region is entered; and~~

commencing work in the first critical region in response to an indication that the first critical region was not previously interrupted.

15. (Previously Presented) The method of Claim 14, further comprising:

setting a flag to indicate that critical region work is not being performed in response to the critical region work completing; and

resetting a counter of times the first critical region was interrupted to zero.

16. (Currently Amended) A method comprising:

disabling interrupts of a critical region in response to the critical region having been interrupted more than a permitted number of times; ~~and~~

commencing critical region work~~, wherein the thread is to avoid disabling all interrupts before it enters the first critical region and avoid re-enabling all interrupts after exiting the first critical region~~;

setting a flag to indicate that critical region work is not being performed in response to the critical region work completing;

resetting a counter of times the critical region was interrupted to zero in response to the critical region work completing;

re-enabling interrupts of the critical region;

interrupting a thread in the critical region;

setting the thread to restart at a beginning of the critical region in response to an indication that the thread is working in a critical region, wherein setting the thread to start at the beginning of the critical region comprises setting a stack pointer and program counter so that the interrupted thread restarts at the beginning of the critical region;

saving stack conditions of an interrupted critical region in response to the critical region not having been previously interrupted, wherein a thread that is interrupted is to avoid disabling all interrupts before it enters the critical region and avoid re-enabling the interrupts after exiting the critical region;

setting a flag to indicate that the critical region is entered;

performing critical region interrupt recovery work in response to an indication that the critical region was previously interrupted; and

commencing work in the critical region.

17-19. (Canceled)

20. (Currently Amended) The method of Claim [[19]] 16, further comprising:

setting a flag to indicate that critical region work is not being performed in response to the critical region work completing; and

resetting a counter of times the critical region was interrupted to zero in response to the critical region work completing.

21. (Currently Amended) An article comprising a non-transitory storage medium, the storage medium comprising machine readable instructions stored thereon when executed by a processor to:

interrupt a thread during a first critical region with an interrupt routine; and

set the thread to restart at a beginning of the first critical region in response to an indication that the thread operates in a critical region, wherein the thread is to avoid disabling all interrupts before it enters the first critical region and avoid re-enabling the interrupts after exiting the first critical region wherein the instructions to set the thread to restart is to comprise instructions to:

set a stack pointer and program counter so that the interrupted thread restarts at the beginning of the first critical region;

increment a count of a number of times the first critical region has been interrupted in response to the first critical region having been previously interrupted; and

save program stack conditions of the interrupted thread in response to the thread not having been interrupted during the first critical region more than a maximum permitted number of times;

set a flag to indicate that a critical region is entered;

perform critical region interrupt recovery work in response to an indication that the first critical region was previously interrupted; and

attempt to complete first critical region work.

22-24. (Canceled)

25. (Currently Amended) The article of Claim [[24]] 21, wherein the critical region interrupt recovery work includes instructions to correct errors that arise from previously interrupting the first critical region prior to its completion.

26. (Currently Amended) The article of Claim [[24]] 21, wherein the first critical region further includes instructions to:

set a flag to indicate that critical region work is not being performed in response to the critical region work completing; and

set to zero the count of the number of times the first critical region was interrupted.

27. (Original) The article of Claim 21 further comprising instructions to mark the thread as having been interrupted during the first critical region.

28. (Original) The article of Claim 21, further comprising instructions to indicate that the thread is not operating in a critical region.

29. (Currently Amended) An apparatus comprising:

an I/O controller device comprising a processor and a memory device, the I/O controller device comprising logic to:

interrupt a thread during a first critical region with an interrupt routine; and

set the thread to restart at a beginning of the first critical region in response to an indication that the thread operates in a critical region, wherein the thread is to avoid disabling all interrupts before it enters the first critical region and avoid re-enabling the interrupts after exiting the first critical region, wherein the logic to set the thread to restart is to comprise logic to: set a stack pointer and program counter so that the interrupted thread restarts at the beginning of the first critical region;

increment a count of a number of times the first critical region has been interrupted in response to the first critical region having been previously interrupted;

save program stack conditions of the interrupted thread in response to the thread not having been interrupted during the first critical region more than a maximum permitted number of times;

set a flag to indicate that a critical region is entered;

perform critical region interrupt recovery work in response to an indication that the first critical region was previously interrupted; and

attempt to complete first critical region work.

30-32. (Canceled)

33. (Currently Amended) The apparatus of Claim [[32]] 29, wherein the critical region interrupt recovery work includes logic to correct errors that arise from previously interrupting the first critical region prior to its completion.

34. (Currently Amended) The apparatus of Claim [[32]] 29, wherein the first critical region further includes logic to:

set a flag to indicate that critical region work is not being performed in response to the critical region work completing; and

set to zero the count of the number of times the first critical region was interrupted.

35. (Original) The apparatus of Claim 29 further comprising logic to mark the thread as having been interrupted during the first critical region.

36. (Original) The apparatus of Claim 29, further comprising logic to indicate that the thread is not operating in a critical region.

37. (Currently Amended) A system comprising:

a first storage subsystem and a second storage subsystem;

a first circuit card including first circuitry capable of being coupled to the first storage subsystem, wherein the first circuitry further includes an I/O controller device, the I/O controller device comprising:

a processor;

a memory device, wherein the memory device includes code segments that instruct the processor to:

interrupt a thread during a first critical region with an interrupt routine, and

set the thread to restart at a beginning of the first critical region in response to an indication that the thread operates in a critical region; and

a second circuit card including second circuitry capable of being coupled to the second storage subsystem, wherein the thread is to avoid disabling all interrupts before it enters the first critical region and avoid re-enabling the interrupts after exiting the first critical region, wherein code is to instruct the processor to: set a stack pointer and program counter so that the interrupted thread restarts at the beginning of the first critical region;

increment a count of a number of times the first critical region has been interrupted in response to the first critical region having been previously interrupted;

save program stack conditions of the interrupted thread in response to the thread not having been interrupted during the first critical region more than a maximum permitted number of times;

set a flag to indicate that a critical region is entered;

perform critical region interrupt recovery work in response to an indication that the first critical region was previously interrupted; and

attempt to complete first critical region work.

38. (Original) The system of Claim 37, wherein the second circuit includes an I/O controller device.

39. (Original) The system of Claim 37, wherein the first storage subsystem and the second storage subsystem each comprise one or more respective mass storage devices.

40. (Original) The system of claim 37, wherein:

the first storage subsystem comprises a redundant array of inexpensive disks (RAID); and

the second storage subsystem comprises a tape mass storage system.

41. (Original) The system of claim 37, further comprising:

a circuit board coupled to the bus, wherein the circuit board comprising a bus, memory, and a host processor; and

the first circuit card and the second circuit card are capable of being coupled to the bus.

42. (Original) The system of Claim 41, wherein the bus complies with PCI.

Conclusion

Any inquiry concerning this communication or earlier communications from the examiner should be directed to ABDULLAH AL KAWSAR whose telephone number is (571)270-3169. The examiner can normally be reached on 7:30am to 5:00pm, EST.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Meng Ai T. An can be reached on 571-272-3756. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

/Meng-Ai An/
Supervisory Patent Examiner, Art Unit 2195

/Abdullah-Al Kawsar/
Examiner, Art Unit 2195